15-740 Course Project Milestone Report

Dong Zhou, Mu Li

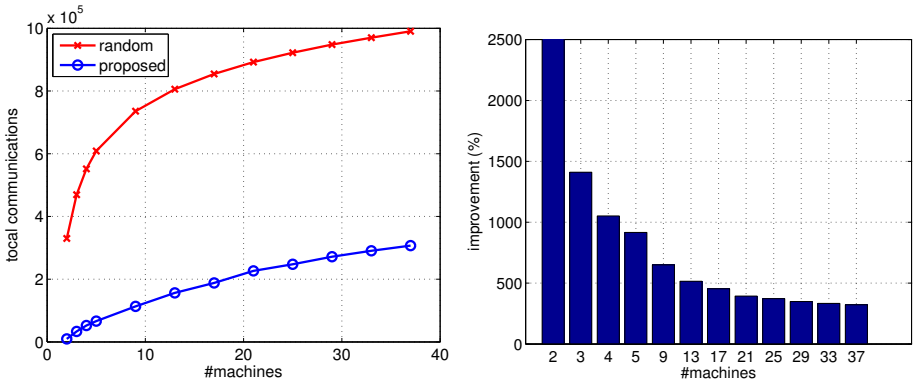
November 19, 2012

Major Changes: Our initial goal is to implement efficient sparse matrix vector multiplication by multi-threading, reducing cache miss, and taking advantage of NUMA. Different to previous works, we do not assume the sparse matrix could be fitted into memory. We use datasets from Internet companies, and found that they have special structures: the numbers of nonzero entries on each columns obey power law distribution. We call it "scale-free" as their according graphs have scale-free property.

Besides traditional technologies, most of which have been studied by previous works, we want to explore how to use scale-free property to accelerate computing. One thing we have tried is to transform the original sparse matrix to a near block diagonal form, which could reduce the amount of variables shared by different threads, cores, and processors. It works pretty well on scale-free matrix. So we would like to propose efficient transform algorithms.

What You Have Accomplished So Far: Things we have done are mainly

- Pre-processing the raw data. Our raw dataset consists of (key, value) pairs and is of 600 Gigabyte size. We used Hadoop to convert it into compressed sparse row format.
- Proposed an transform algorithm to re-order both columns and rows of a sparse matrix to make it as block diagonal as possible. Comparing to the original matrix, the re-ordered matrix reduces inter threads (or processors) communications significantly, which is shown on the following figures.



(a) total communications versus number of processors

(b) improvements versus numbers of processors

Meeting Your Milestone: We have finished the 75% goal in our proposal, though NUMA-aware optimizations are not finished yet. The main reason is that pre-processing our large dataset costs more time than we expected: (1) It costs a few days for us to get access to a Hadoop cluster and upload our dataset to the HDFS (2) We need a distributed hash table but it's not easy to be implemented on Hadoop.

The most surprising finding is that we can reduce the inter-processor traffic significantly by properly re-ordering rows and columns of our matrices. This technology has not been studied much by others so far, so we will focus our effort on this point.

Revised Schedule: On the following two weeks, we plan to

Weak 1: Implement a disk-based sparse matrix transform algorithm;

Weak 2: Explore the performance gain by using the near block diagonal form.

Resources Needed: None.